Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently amended) A method of scheduling access of a table by multiple processes, comprising:

in a software-implemented procedure, associating a lock level with a particular process, a higher lock level representing a larger number of other processes having priority over the particular process in accessing the table, each of the processes being associated with no more than one lock level;

the particular process repeatedly attempting to associate the particular process with a lower lock level, and if the particular process has been successfully associated with the lower lock level, releasing a previous lock level associated with the particular process so that the previous lock level may be associated with other processes; and

the software-implemented procedure allowing the particular process to access the table when the lock level for the particular process is equal to a preset value.

2. (Original) The method of claim 1 in which the preset value is equal to one.

3. (Original) The method of claim 1 in which each of the processes attempts to associate itself with a lower lock level independently of other processes.

4. (Original) The method of claim 1, further comprising storing in a queue information indicating which process is associated with which lock level.

5. (Currently amended) The method of claim 1, further comprising calling multiple instances of [[a]] the procedure that associates a lock level with a process, each instance of the procedure associated with one of the multiple processes and [[is]] configured to attempt to associate a different lock level with the process associated with the instance until the process is granted access to the record.

6. (Original) The method of claim 1, further comprising allowing processes to read the record but not modify the record when the lock levels for the processes are different from the preset value.

7. (Original) The method of claim 1, further comprising locking the record when the lock level having the preset value is associated with a process.

8. (Original) The method of claim 1 in which at least two of the processes are being run in a parallel processing environment.

9. (Currently amended) A method comprising:

in a software-implemented procedure,

upon receiving a request from a first process to access a record in a database, associating a first lock level [[to]] with the first process and allowing the first process to access the record, preventing other processes from ~~accessing~~ modifying the record until the first process finishes accessing the record;

upon receiving a request from a second process to access the record while the first process is still accessing the record, associating a second lock level [[to]] with the second process;

upon receiving a request from a third process to access the record while the first process is still accessing the record, associating a third lock level with the third process;

when the first process finishes accessing the record, releasing the first lock level; and

the second and third processes each repeatedly attempting to associate itself with a lower lock level, and when one of the second and third processes associates itself with the first lock level, permitting the process to modify the record.

~~either (a) releasing the second lock level from being associated with the second process and associating the first lock level with the second process, allowing the second process to access the record but preventing other processes from accessing the record until the second process finishes accessing the record, and when the second process finishes accessing the record, releasing the first lock level from being associated with the second process, or~~

~~(b) releasing a lock level from being associated with a third process and associating the first lock level with the third process, allowing the third process to access the record but preventing other processes from accessing the record until the third process finishes accessing the record, and when the third process finishes accessing the record, releasing the first lock level from being associated with the third process.~~

10. (Currently amended) The method of claim 9 in which preventing other processes from ~~accessing~~ modifying the record comprises allowing the other processes to read the record but not modify the record.

11. (Original) The method of claim 9, further comprising locking the record when the first lock level is associated with a process.

12. (Original) The method of claim 8, further comprising writing to a queue to specify which lock level is associated with which process.

13. (Original) The method of claim 9 in which at least two of the first, second, and third processes are being run in a parallel processing environment.

14. (Currently amended) A method comprising:

in a software-implemented procedure,

locking a record in a database at multiple levels when multiple processes running in parallel attempt to access the record;

assigning a lock level to each of the multiple processes, ~~different processes~~ each process having a different lock level, ~~levels;~~ each of the processes not currently associated with a lowest lock level repeatedly attempting to associate itself with a lower lock level; and

selectively permitting one of the multiple processes to access the record at a time.

15. (Original) The method of claim 14, further comprising reassigning the lock levels of the processes when a process accessing the record terminates its access to the record.

16. (Original) The method of claim 15 in which a process that attempted to access the record earlier than another process is assigned a lower lock level than the other process, and each process other than the process terminating its access to the record is assigned a lower lock level when the process terminates its access to the record.

17. (Original) The method of claim 14, further comprising storing in a queue information indicating which process is associated with which lock level.

18. (Original) The method of claim 14, further comprising calling multiple instances of a procedure that assigns a lock level to a process, each instance of the procedure associated with one of the multiple processes and is configured to attempt to assign a different lock level to the process until the process is granted access to the record.

19. (Currently amended) A system comprising:

a database to store records; ~~and~~

a queue to store information relating to lock levels of processes that attempt to access the records, each different ~~processes~~ process having a different lock ~~levels~~ level when ~~accessing~~ attempting to access the same record, one of the processes having a particular lock level, the process having the particular lock level being allowed to access the record; and

a programmable processor to execute a procedure to assign a respective lock level to each of the different processes, each of the different processes having a lock level other than the particular lock level repeatedly attempting to associate itself with another lock level that is closer to the particular lock level, the procedure allowing any one process to access the record when that one process has the particular lock level.

20. (Original) The system of claim 19, further comprising a memory to store software code for implementing a procedure in which instances of the procedure are used to assign lock levels to the processes.

21. (Original) The system of claim 19 in which the software code is configured so that the instances of the procedure are run in parallel.

22-25. (Cancelled)

26. (Currently amended) A computer program product, tangibly stored on a machine-readable medium, for implementing a multi-level lock process, comprising instructions operable to cause one or more programmable processors to:

upon receiving a request from a first process to access a record in a database, associate a first lock level [[to]] with the first process and allow the first process to access the record but prevent other processes from accessing the record until the first process finishes accessing the record;

upon receiving a request from a second process to access the record while the first process is still accessing the record, associate a second lock level [[to]] with the second process;

upon receiving a request from a third process to access the record while the first process is still accessing the record, associate a third lock level with the third process;

when the first process finishes accessing the record, release the first lock level from being associated with the first process; [[,]] and

cause each of the second and third processes to independently attempt to associate itself with a lower lock level, and when one of the second and third processes associates itself with the first lock level, permitting the process to access the record.

~~either (a) release the second lock level from being associated with the second process and associate the first lock level with the second process, allowing the second process to access the record but preventing other processes from accessing the record until the second process finishes accessing the record, and when the second process finishes accessing the record, releasing the first lock level from being associated with the second process, or~~

~~(b) release a lock level from being associated with a third process and associate the first lock level with the third process, allowing the third process to access the record but preventing other processes from accessing the record until the third process finishes accessing the record, and when the third process finishes accessing the record, releasing the first lock level from being associated with the third process.~~

27. (Original) The computer program product of claim 26 in which the instructions cause the one or more programmable processors to allow the other processes to read the record but not modify the record.

28. (Original) The computer program product of claim 26, further comprising instructions operable to cause the one or more programmable processors to lock the record when the first lock level is associated with a process.

29. (Original) The computer program product of claim 26, further comprising instructions operable to cause the one or more programmable processors to write to a queue to specify which lock level is associated with which process.

30. (Original) The computer program product of claim 26, in which the instructions cause the one or more programmable processors to execute at least two of the first, second, and third processes in a parallel processing environment.

31. (Currently amended) A computer program product, tangibly stored on a machine-readable medium, for scheduling access of a table by multiple processes, comprising instructions operable to cause one or more programmable processors to:

associate a lock level with a particular process, a higher lock level representing a larger number of other processes having priority over the particular process in accessing the table, each of the processes being associated with no more than one lock level;

repeatedly attempt to associate the particular process with a lower lock level, and if the particular process has been successfully associated with the lower lock level, release a previous lock level associated with the particular process so that the previous lock level may be associated with other processes; and

allow the particular process to access the table when the lock level for the particular process is equal to a preset value.

32. (Original) The computer program product of claim 31 in which the instructions are configured so that the preset value is equal to one.

33. (Original) The computer program product of claim 31 in which the instructions are configured so that each of the processes attempts to associate itself with a lower lock level independently of other processes.

34. (Currently amended) A computer program product, tangibly stored on a machine-readable medium, for implementing a multi-level lock process, comprising instructions operable to cause one or more programmable processors to:

lock a record in a database at multiple levels when multiple processes running in parallel attempt to access the record;

assign a lock level to each process, different processes having different lock levels;

cause each of the processes to repeatedly attempt to associate itself with a lower lock level; and

selectively allow one of the multiple processes to access the record at a time.

35. (Original) The computer program product of claim 34 further comprising instructions operable to cause the one or more programmable processors to reassign the lock levels of the processes when a process accessing the record terminates its access to the record.

36. (Original) The computer program product of claim 34 in which the instructions are configured so that a process that attempted to access the record earlier than another process is assigned a lower lock level than the other process, and each process other than the process terminating its access to the record is assigned a lower lock level when the process terminates its access to the record.

37. (Original) The computer program product of claim 34 further comprising instructions operable to cause the one or more programmable processors to store in a queue information indicating which process is associated with which lock level.

38. (Original) The computer program product of claim 34, further comprising instructions operable to cause the one or more programmable processors to call multiple instances of a procedure that assigns a lock level to a process, each instance of the procedure associated with

one of the multiple processes and is configured to attempt to assign a different lock level to the process until the process is granted access to the record.

39. (New) The method of claim 1 wherein the different processes operate in parallel to attempt to associate with lower lock levels.

40. (New) The method of claim 31 wherein the different processes operate in parallel to attempt to associate with lower lock levels.